Solving an SVM by hand based on KKT conditions

by Forrest Sheng Bao at Iowa State University

Although necessary but not sufficient, the KKT conditions are sufficient for optimality if the problem is to minimize a convex function which is the case for an SVM in primal form. Therefore, from KKT conditions alone, we can solve an SVM by hand.

Solving a system of equations in Mathematica

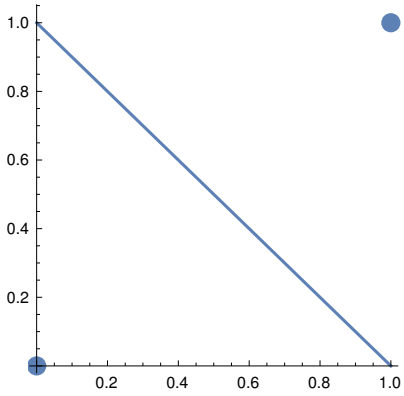
In[ ]:=
```
Solve[{a1 + a2 == 2, a1 - a2 == 10}, {a1, a2}]
```

Out[ ]=
```
{{a1 → 6, a2 → -4}}
```

Solving a simple SVM of only two samples.

The plot below indicates two two samples, (1,1) of class +1, and (0,0) of class -1. Denote the first one as sample 1 and the second as sample 2. A well-trained SVM would be the one depicted by the blue line.

In[ ]:=
```
plt1 = ListPlot[{{0, 0}, {1, 1}}, PlotStyle → {AbsolutePointSize[10]}, AspectRatio → 1];
plt2 = ListLinePlot[{{0, 1}, {1, 0}}];
Show[plt1, plt2]
```



1. First, using the gradient on the bias term that $\sum_{k=1}^{K} \lambda_k \, yk = 0$, we have $(\lambda_1 \quad \lambda_2) \begin{pmatrix} +1 \\ -1 \end{pmatrix} = 0$

2. Then, write one equation for each sample in the complementary slackness equation (Equation D in slides): $\lambda_1\left[+1\cdot\left(\begin{pmatrix} w1 & w2 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix}+w_b\right)-1\right]=0$

   $\lambda_2\left[-1\cdot\left(\begin{pmatrix} w1 & w2 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \end{pmatrix}+w_b\right)-1\right]=0$

3. Finally, the gradient on the non-bias weights: $\begin{pmatrix} w1 \\ w2 \end{pmatrix}=\lambda_1\cdot\begin{pmatrix} 1 \\ 1 \end{pmatrix}-\lambda_2\cdot\begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Expand matrix equations above into a system of linear equations and solve them using the Solve function. For typesetting easiness, λ's are replaced with a's in the code below.

In[ ]:=
```
Solve[{ a1 - a2 == 0,
        a1 * (w1 + w2 + wb - 1) == 0,
        a2 * (-1 * wb - 1) == 0,
        w1 == a1,
       w2 == a1
 }, {a1, a2, w1, w2, wb}
]
```

Solve: Equations may not give solutions for all "solve" variables.

Out[ ]= {{a1 → 0, a2 → 0}, {a1 → 0, a2 → 0, wb → -1}, {a1 → 0, a2 → 0, wb → 1}, {a1 → 1, a2 → 1, wb → -1}}

The first 3 solutions set both a1 and a2 to 0. They should be discarded as they remove the constraints from the Lagrange multiplier.
The last solution is what we want, demonstration an equation $x + y - 1 = 0$ which beautifully goes between the two samples at 135 degree. The fact that neither a1 nor a2 are zero indicates that both samples are the supporting vectors.

Let's see more examples.

In[3]:=
```
Solve[{a1 + a2 - a3 - a4 == 0,
       a1 * (1 * (w1 + w2 + wb) - 1) == 0,
       a2 * (1 * (w1 + 0 + wb) - 1) == 0,
       a3 * (-1 * (0 + 0 + wb) - 1) == 0,
       a4 * (-1 * (0 + w2 + wb) - 1) == 0,
       w1 == a1 + a2,
       w2 == a1 - a4,
  a1 ≥ 0, a2 ≥ 0, a3 ≥ 0, a4 ≥ 0
 }, {w1, w2, a1, a2, a3, a4, wb}
]
```

Solve: Equations may not give solutions for all "solve" variables.

Out[3]=
```
{{w1 → 0, w2 → 0, a1 → 0, a2 → 0, a3 → 0, a4 → 0},
 {w1 → ConditionalExpression[2, 0 < a4 < 2], w2 → ConditionalExpression[0, 0 < a4 < 2],
  a1 → ConditionalExpression[a4, 0 < a4 < 2], a2 → ConditionalExpression[2 - a4, 0 < a4 < 2],
  a3 → ConditionalExpression[2 - a4, 0 < a4 < 2], wb → ConditionalExpression[-1, 0 < a4 < 2]},
 {w1 → 1, w2 → -1, a1 → 0, a2 → 1, a3 → 0, a4 → 1, wb → 0},
 {w1 → 1, w2 → 1, a1 → 1, a2 → 0, a3 → 1, a4 → 0, wb → -1},
 {w1 → 2, w2 → 0, a1 → 0, a2 → 2, a3 → 2, a4 → 0, wb → -1},
 {w1 → 2, w2 → 0, a1 → 2, a2 → 0, a3 → 0, a4 → 2, wb → -1}}
```

In[ ]:=
```
Solve[{a1 - a2 - a3 + a4 == 0,
       a1 * (1 * (w1 + w2 + wb) - 1) == 0,
       a2 * (-1 * (w1 + 0 + wb) - 1) == 0,
       a3 * (-1 * (0 + 0 + wb) - 1) == 0,
       a4 * (1 * (0 + w2 + wb) - 1) == 0,
       w1 == a1 - a2,
       w2 == a1 + a4,
  a1 ≥ 0, a2 ≥ 0, a3 ≥ 0, a4 ≥ 0
 }, {w1, w2, a1, a2, a3, a4, wb}
]
```

Solve: Equations may not give solutions for all "solve" variables.

Out[ ]=
```
{{a1 → 0, a2 → 0, a3 → 0, a4 → 0}, {a1 → 1, a2 → 0, a3 → 1, a4 → 0, wb → -1}}
```

Get: Cannot open CloudObjectLoader`.